AMATH 582 Final Project

William Livernois (willll@uw.edu)

March 19, 2020

Abstract

Quantum transport through a DNA structure was monitored as a function of the contact self-energies. Looking at principle components of the transmission spectrum and applying regularization methods it was possible to determine the correlation between the self-energies of the contacts and the transmission energy levels present in the structure.

1 Introduction and Overview

Quantum transport through a double-stranded DNA strand was modeled using a combination of density functional theory (DFT) and non-equilibrium Green's function (NEGF) methods. The DFT used a Gaussian-type orbital basis to solve the Hartree-Fock equations and generate a Hamiltonian matrix for the isolated system. This Hamiltonian was modified to calculate quantum transport using the NEGF formulation based on contact coupling parameters, called self-energies and represented by Γ_L and Γ_R for the left and right contacts of the two-terminal device. These self-energies represent electron hopping rates in the NEGF transport calculations and are dependent on the geometry and material properties of the contact. The result of these transport calculations is a transmission spectrum (electronic wavefunction transmission as a function of energy) that can be used to generate a current/voltage characteristic for the device and measure low-bias conductance.

In this dataset the self energies for the left and right contacts were varied to examine the effect on the calculated transmission through the DNA strand, which is shown in Figure 1. A total of 15 values were chosen for each contact self-energy, giving a total data set of 225 (15^2) transmission spectra. The transmission spectra were calculated on an energy grid of 600 points between -6eV and 0eV, which are relative energies based on the basis set used in the DFT calculations. This energy range is centered around

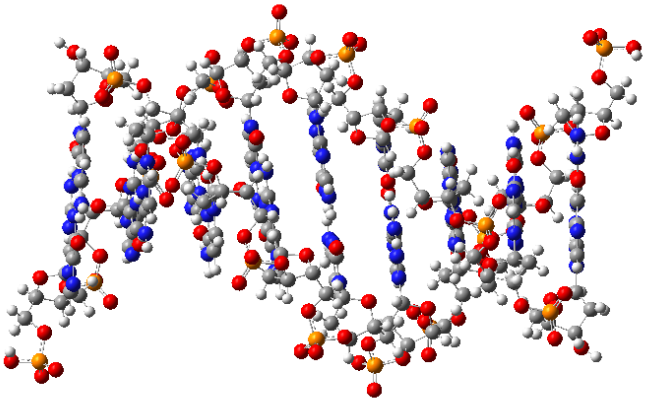

Figure 1: Structure of the double-stranded DNA strand showing 9 base-pairs and color-coded atoms (N=blue, C=gray, H=white, P=orange, O=red)

the highest occupied molecular orbital (HOMO) and lowest unoccupied molecular orbital (LUMO) energies which were calculated to be -4.7620 eV and -0.7347 eV respectively. Principle Components Analysis was used to determine the major modes of variance within the transmission spectrum, giving an overall picture of the transmission as a function of the contact self-energies. Various regression methods using L1 and L2 regularization were applied to map the transmission spectra to the contact self-energies. These coefficients represent the weight of each energy level on the contact transmission, which can be used to provide further information about electronic structure and transport properties.

2 Theoretical Background

For a stationary periodic system, DFT solves for the wavefunctions and energy levels of the electrons directly from the system potential. While this information can be used to determine electronic properties of bulk materials, boundary effects that change transport behavior become more relevant for nanoscale systems. For this study the Gaussian 16 software package[1] was used to find the energy levels (eigenvalues) and bound states (eigenvectors describing the combination of atomic orbitals) of the initial isolated system. Then, the NEGF method can be applied using the contact self-energy parameters to model the coupled system with the electric contacts [2]. In this set of experiments the contacts were applied to each end of the DNA strand and the self-energy of each of these contacts (Γ_L and Γ_R) were modified to generate the various transmission spectra. These were combined together into a transmission matrix for this analysis.

Principle component analysis was conducted on the transmission matrix using singular value decomposition (SVD). The transmission matrix T can be separated into the product $T = U\Sigma V^T$, with rotation matrices U and V^T, and diagonal singular value matrix Σ. Using this transformation, the orthogonal modes of variance could be separated for analysis, each mode (columns of U) with an energy weight given by the corresponding Σ value. For this analysis, the coefficients for the first dominant mode were used to determine the impact of the contact self-energies on the overall transmission and the top 23 principle components (accounting for for ¿99% of the variance) were used in an L1 optimization.

Four methods were used to apply a regression to the transmission and self-energy data. The system is highly under-determined (601 energy points versus 225 self energy combinations) and therefore requires regularization methods to determine coefficients. In addition to the built-in solvers (whose algorithms minimize an L2 norm), two additional functions were used that implemented an L1 norm minimization in order to promote sparsity in the coefficient vector. This optimization problem can be defined with a general curve fitting function that fits points x_i, y_i to some formula $y(x)$. The coefficients are determined by minimizing an error E given by

$$E_p = \left(\frac{1}{N} \sum_{n=1}^{N} |y(x_i) - y_i|^p \right)^{1/p} \tag{1}$$

where p=1 corresponds to the L1 norm and p=2 corresponds to the L2 norm [3]. To promote sparsity of the coefficients, the L1 norm is preferred because this allows the coefficients to be concentrated around important features rather than being distributed across the whole data set. In our dataset, this sparsity corresponds to peaks at certain energies that show significant correlation with the self-energies.

3 Algorithm Implementation and Development

Principal components were extracted using `svd()` MATLAB function that uses a proprietary algorithm that is based on the `DGESVD` function in the LAPACK library[4]. Transmission matrix was constructed by concatenating each spectrum as a rows in a matrix and storing the corresponding Γ_L and Γ_R combinations in column vectors. The transmission spectra ranged multiple orders of magnitude (usually it is shown in a log plot for visualization) and any reconstruction with negative transmission values are not physical. Therefore, the negative modes and coefficients were swapped (multiplied by -1) for visualization. Reconstruction of the transmission modes was possible with only a few modes, and the same low-rank U matrix was stored for

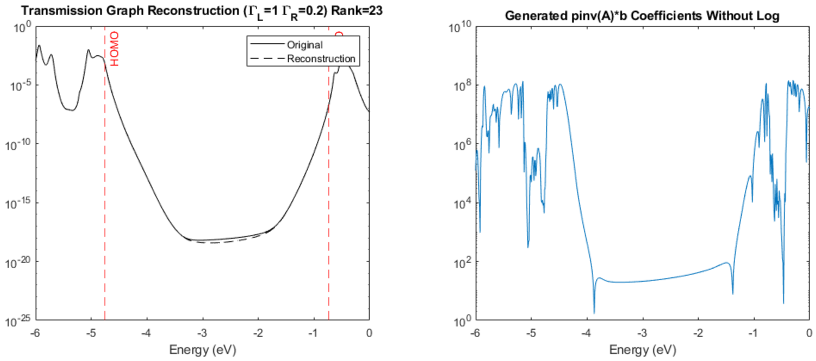

Figure 2: A comparison of one spectrum with the PCA reconstruction (left) and regression coefficients calculated from the unmodified transmission data

future analysis.

The large range of transmission values was especially an issue for applying a regression to pinpoint energies. Using the unmodified values resulted in energy weights that almost exactly matched the original transmission spectrum, especially for regressions that minimize an L2 norm. A comparison of one spectrum (as well as it's PCA reconstruction) and some generated regression coefficients from the unmodified transmission values are shown in Figure 2 demonstrating the weighting issue (especially with the L2 norm). Therefore, for three of the applied regression techniques the log of the transmission data was used. The first regression method used was the `mldivide` MATLAB function (more commonly shown as `A\b`) which uses QR factorization to minimize the L2 norm. The second method used the `pinv` MATLAB function, which solves for the Moore-Penrose pseudo inverse that also minimizes the L2 norm. Two L1 norm techniques were applied to compare to these results: minimization of the L1 norm of the low-rank transformed (overdetermined) system and the least absolute shrinkage and selection operator (LASSO) method applied to the full under-determined system. For the first method, the `fminsearch` was used to minimize the sum of L1 norms using the principle components of the system, and the resulting coefficients were transformed back to the energy basis for comparison to the other methods. The initial conditions were set by the Σ values to allow for representative weighting in the low-rank basis. By contrast, the LASSO method minimizes the the L2 norm of the error with an added regularization parameter that controls the weight of the L1 norm of the coefficient, with the full objective given by

$$min\left(\sum_n (|Ax_n - b|^2 + \lambda|x_n|)\right) \tag{2}$$

Several values of λ were compared based on the outputs from `lasso` MATLAB function with very similar results, though a final value of $\lambda = 0.001$ that was used for this study. A final regression method was

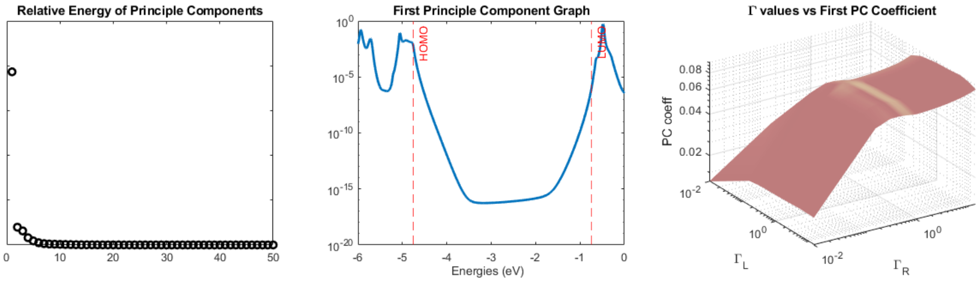

Figure 3: A comparison of the energies of the first 50 SVD modes (left) with the dominant mode plotted (middle) and the resulting coefficients graphed as a function of Γ_L and Γ_R (right)

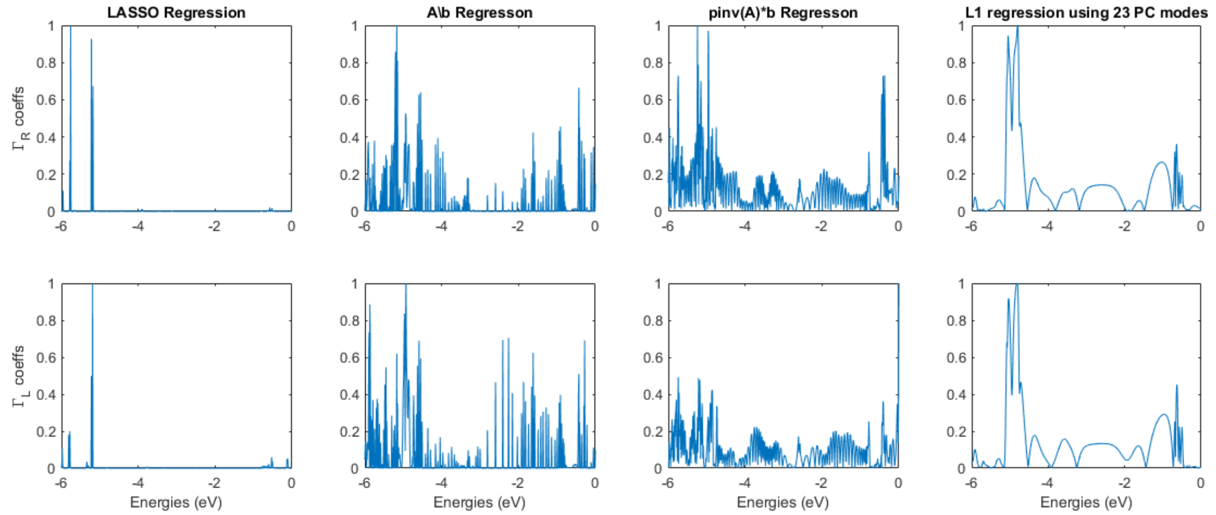

Figure 4: A comparison of the scaled coefficients generated from each of the four regression methods for the left and right contact self-energies.

attempted using the CVX MATLAB package[5], but a solution could not be found because the dataset was not convex, even with the log transformation and change of basis.

4 Computational Results

The relative energies of the first 50 principle components are shown in Figure 3 along with a graph of the dominant mode and resulting coefficients for each pair of gamma values (displayed as a surface plot). This result clearly shows that there is one dominant mode that accounts for 70% of the variance, and the surface plot gives us some insight into the effect of the self-energies on the overall transmission. At lower Γ values, the transmission increases steadily up to a point, followed by some non-linearity above a certain value. This can be attributed to a maximization of the ”coupling” meaning that the quantum transport is limited by the system itself and not by the interaction with the contacts.

The resulting regression coefficients are shown in Figure 4 from each of the four regression methods. It is clear from these results that the L1 norm minimization helped to increase the sparcity of coefficients, especially with the LASSO result. The rank reduction with the principle component modes to minimize the L1 norm was an improvement over the L2 norm methods but the matrix transformation back to the energy basis caused peak broadening and did not lead to any interesting conclusions about the system. Looking at the residuals of the different regressions it was possible to see how well the coefficients reproduced the

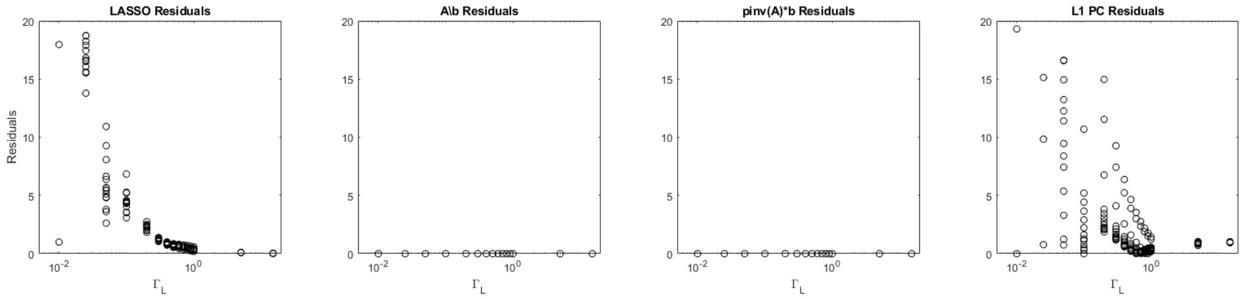

Figure 5: A comparison of the residuals for the Γ_L regression from each of the four methods used.

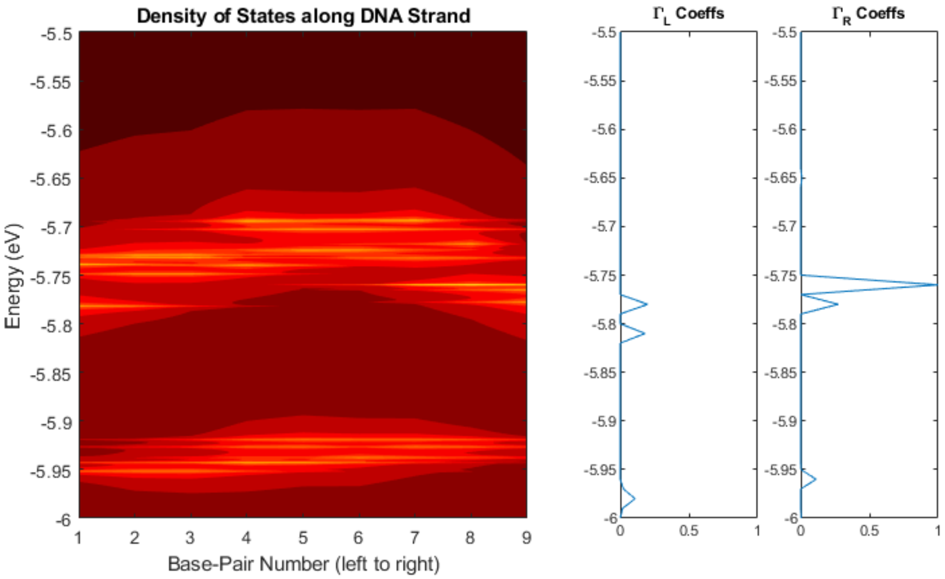

Figure 6: A comparison of the 2D DOS plot peaks with LASSO regression coefficients, with peak matching especially apparent around -5.75 eV

expected Γ factors. As can be seen in Figure 5, both L2 norm regressions had very low overall residuals whereas the L1 norm regressions had rather high residuals especially at lower Γ values. This tells us that though the L1 norm methods were useful for producing a more sparse set of coefficients the resulting fit was more applicable at higher values of Γ.

5 Summary and Conclusions

While the results from the PCA provided some interesting insight into the modes of variability of the transmission, the results from the LASSO regression was useful for pinpointing specific energies that were impacted by the Γ values. Furthermore, the regression had different results for the left and right contact, which can be compared to locations on the physical structure. From the initial DFT calculation a density of states (DOS) plot was generated as a function of location, represented by weight of orbitals located at each base pair. The DOS energy peaks at the left and right contact actually line up very closely with the weights generated from the L1 regression, as shown in Figure 6, demonstrating the local coupling between the contact and DNA structure as given by the Γ contact self-energy value. It makes sense that these peaks dominate in the regression, and looking at the other weights can give some insight into other energy states that participate in quantum transport at the contacts. It is important to note that the overall regression results were fitted to higher Γ values, and repeating this regression with a more evenly distributed grid could be use to generate more accurate results.

In a future study it would be illuminating to use PCA to compare the transmission plots for several different physical systems. In this study one (unchanged) structure was used and this lead to only one major mode of variability. Comparing multiple strands (perhaps with a different base-pair sequence) or different contact sites with PCA would provide some information about common transmission modes for the data. Furthermore, the regression methods explored here could be used to determine important states that participate in quantum transport. These energies correspond to states represented by actual electronic wave functions, which can be used to measure orbital location and local electron behavior.

References

[1] M. J. Frisch et al. *Gaussian 16 Revision C.01.* Gaussian Inc. Wallingford CT. 2016.

[2] Supriyo Datta. "Nanoscale device modeling: the Green's function method". In: *Superlattices and Microstructures* 28.4 (2000), pp. 253–278. ISSN: 0749-6036. DOI: `https://doi.org/10.1006/spmi.2000.0920`.

[3] Jose Nathan Kutz. *Data-driven modeling & scientific computation: methods for complex systems & big data.* Oxford University Press, 2013.

[4] Edward Anderson et al. *LAPACK users' guide.* SIAM, 1999.

[5] Michael Grant and Stephen Boyd. *CVX: Matlab Software for Disciplined Convex Programming, version 2.1.* `http://cvxr.com/cvx`. Mar. 2014.

Appendix A MATLAB Functions

The MATLAB functions used for these calculations are listed below:

- `vals = regexp(str, pattern, 'match')` returns an array of strings `vals` from `str` that match the regular expression `pattern`
- `[X1, ... ,XN] = ndgrid(x1, ... , x2)` returns n-dimensional grid matrices with coordinates that span the values contained in the vectors `x1` through `xn`
- `vq = griddata(x, y, v, xq, yq)` interpolates data given by `(x, y, v)` to the grid given by `xq` and `xq`
- `[u, s, v] = svd(X)` returns the rotation matrices `u` and `v` and the diagonal matrix `s` as the singular value decomposition of `X` such that `X=u*s*v'`
- `m = diag(M)` returns the diagonal vector `m` of matrix `M`
- `x = A\b` runs an L2 regression on `Ax = b` to generate coefficients `x`
- `B = pinv(A)` calculates the Moore-Penrose Pseudoinverse of matrix `A`
- `x = fminsearch(func, x0, opt)` minimizes function `func` with initial conditions `x0` and optimization parameters `opt` that can be set using the `optimset(Param, value)` method
- `x = lasso(A, b, 'Lambda', lambda)` runs a LASSO regression on `Ax=b` with L1 regularization component λ given by `lambda`

Functions used for basic arithmetic or plot formatting are not included above, but the important figures are included in the Computation Results section above.

Appendix B MATLAB Code

All code is located in the ”Final Project” folder of the github repository (`https://github.com/wliverno/AMATH582`).

Several MATLAB files were used for this project, the main one being `TransmissionAnalysis.m` which generated the principle components and regressions as well as all visualizations:

```
close all, clear all, clc;

%Import Data
mats =dir('*_BP(E).mat');

gammaL = zeros(length(mats), 1);
gammaR = zeros(length(mats), 1);

E = load(mats(1).name).Energy;
n = length(mats)
T = zeros(length(E), length(mats));

homo= -4.7620;
lumo=-0.7347;

for i = 1:length(mats)
    fname = mats(i).name;
    T(:, i) = load(fname).T;
    gammas = regexp(fname, '_[0-9]+\.*[0-9]*_', 'match');
    gammaLS = cell2mat(gammas(1));
    gammaLS = gammaLS(2:end-1);
    gammaRS = cell2mat(gammas(2));
    gammaRS = gammaRS(2:end-1);
    gammaL(i) = str2double(gammaLS);
    gammaR(i) = str2double(gammaRS);
end
E = load(fname).Energy;

%Get singular modes/values
[U,S,V] = svd(T);

% Plot sigma values and first mode
figure(1);
subplot(1,3,1), plot(diag(S)/sum(diag(S)),'ko','Linewidth',2), axis([0 50 0 1]);
title('Relative Energy of Principle Components');
[GL, GR] = ndgrid(linspace(min(gammaL), max(gammaL), 100), linspace(min(gammaR), max(gammaR), 100));
VV = griddata(gammaL, gammaR, -V(:,1), GL, GR);
subplot(1,3,3), surfl(GL,GR,VV), colormap pink, shading interp;
set(gca, 'XScale', 'log'), set(gca, 'YScale', 'log'), set(gca, 'ZScale', 'log');
xlabel('\Gamma_L'), ylabel('\Gamma_R'), zlabel('PC coeff');
title('\Gamma values vs First PC Coefficient');
subplot(1,3,2), semilogy(E,-U(:,1),'Linewidth',2);
title('First Principle Component Graph');
xline(homo,'--r', 'HOMO','HandleVisibility','off');
xline(lumo,'--r', 'LUMO','HandleVisibility','off');
% subplot(2,2,4), plot(E,U(:,2)/max(abs(U(:,2))),'Linewidth',[2])
% xline(homo,'--r', 'HOMO','HandleVisibility','off');
% xline(lumo,'--r', 'LUMO','HandleVisibility','off');
drawnow;

%Use first 23 components as rank cutoff for PCA
rank = 23;
Ur = U(:, 1:rank); Sr = S(1:rank, 1:rank); Vr = V(:, 1:rank);
Tr = Ur*Sr*Vr';

% Reconstruct tranmission graph using reduced mode
figure(2), subplot(1,2,1);
n_graph = 200
semilogy(E', T(:,n_graph), '-k', E', Tr(:, n_graph), '--k');
title(['Transmission Graph Reconstruction (\Gamma_L=', num2str(gammaL(n_graph)),' \Gamma_R=', num2str(ga
xlabel('Energy (eV)')
axis([-6 0 1e-25 1e0])
legend('Original', 'Reconstruction');
xline(homo,'--r', 'HOMO','HandleVisibility','off');
xline(lumo,'--r', 'LUMO','HandleVisibility','off');


coeffsCompare = pinv(T')*gammaL;
subplot(1,2,2), semilogy(E', abs(coeffsCompare));
title('Generated pinv(A)*b Coefficients Without Log');
xlabel('Energy (eV)');
drawnow;

% Use log of transmission for Regression
LT = log10(T);

% % Use CVX to construct an L1-regression
% [gammaLsort, gammaLind] = sort(gammaL);
% [gammaRsort, gammaRind] = sort(gammaR);
% L1 on GammaL/GammaR
% cvx_setup
% cvx_begin;
%   cvx_solver mosek
%   variable x1(rank);
%   minimize( norm(x1,1) );
%   subject to
%       (T'*Ur)*x1 == gammaLsort;
% cvx_end;
%
% cvx_begin;
%   cvx_solver mosek
%   variable x2(rank);
%   minimize( norm(x2,1) );
%   subject to
%       (T'*Ur)*x2 == gammaRsort;
% cvx_end;
% x1 = Ur*x1;
% x2 = Ur*x2;

% Using built-in matlab A\b solver
x1_ls = LT'\gammaL;
```

```
x2_ls = LT'\gammaR;

% Using pinv to solve
x1_ps = pinv(LT')*gammaL;
x2_ps = pinv(LT')*gammaR;

% Using fminsearch with rank r principle components
[Ul,Sl,Vl] = svd(LT);
Ulr = Ul(:, 1:rank);
Slr = diag(Sl(1:rank, 1:rank));
opt = optimset('MaxFunEvals',10000);
funcL = @(x) sum(abs(LT'*Ulr*x - gammaL));
funcR = @(x) sum(abs(LT'*Ulr*x - gammaR));
x1f_pc=fminsearch(funcL,Slr, opt);
x2f_pc=fminsearch(funcR,Slr, opt);
x1f = Ulr*x1f_pc;
x2f = Ulr*x2f_pc;

% Using LASSO method with Lambda=1e-3
x1 = lasso(LT', gammaL, 'Lambda', 1e-3);
x2 = lasso(LT', gammaR, 'Lambda', 1e-3);

%% Plot comparison of each Method
figure(3);

subplot(2,4,1), plot(E, abs(x2)/max(abs(x2))), title('LASSO Regression'),  ylabel('\Gamma_R coeffs');
subplot(2,4,5), plot(E, abs(x1)/max(abs(x1))),  xlabel('Energies (eV)'), ylabel('\Gamma_L coeffs');
subplot(2,4,2), plot(E, abs(x2_ls)/max(abs(x2_ls))), title('A\b Regresson');
subplot(2,4,6), plot(E, abs(x1_ls)/max(abs(x1_ls))), xlabel('Energies (eV)');
subplot(2,4,3), plot(E, abs(x2_ps)/max(abs(x2_ps))), title('pinv(A)*b Regresson');
subplot(2,4,7), plot(E, abs(x1_ps)/max(abs(x1_ps))),  xlabel('Energies (eV)');
subplot(2,4,4), plot(E, abs(x2f)/max(abs(x2f))), title('L1 regression using 23 PC modes');
subplot(2,4,8), plot(E, abs(x1f)/max(abs(x1f))), xlabel('Energies (eV)');
drawnow;

%% Plot Residuals
figure(4);
clf
hold on
lassoRes = abs(x1'*LT- min(x1'*LT) - gammaL')./gammaL';
backslashRes = abs(x1_ls'*LT - gammaL')./gammaL';
pinvRes = abs(x1_ps'*LT - gammaL')./gammaL';
fminRes = abs(x1f'*LT - gammaL')./gammaL';
subplot(1,4,1), plot(gammaL', lassoRes, 'ko'), set(gca,'xscale', 'log');
axis([0.005 20 0 20]), title('LASSO Residuals'), xlabel('\Gamma_L'), ylabel('Residuals');
subplot(1,4,2), plot(gammaL', backslashRes, 'ko'), set(gca,'xscale', 'log');
axis([0.005 20 0 20]), title('A\b Residuals'), xlabel('\Gamma_L');
subplot(1,4,3),plot(gammaL', pinvRes, 'ko'), set(gca,'xscale', 'log');
axis([0.005 20 0 20]), title('pinv(A)*b Residuals'), xlabel('\Gamma_L');
subplot(1,4,4), plot(gammaL', fminRes, 'ko'), set(gca,'xscale', 'log');
axis([0.005 20 0 20]), title('L1 PC Residuals'), xlabel('\Gamma_L');


% Store Lasso Results
```

```
gammaLcoeffs = abs(x1)/max(abs(x1));
gammaRcoeffs = abs(x2)/max(abs(x2));
save('gammaCoeffs.mat', 'E', 'gammaLcoeffs', 'gammaRcoeffs');
```

The file `DOSPlot.m` generated the 2D DOS plot and compared the LASSO regression coefficients in a figure:

```
%% 2D DOS
clear all, close all, clc;
load DNADOS.mat;
load gammaCoeffs.mat;
N = 9;

temp = zeros(N, length(DOS));
for j = 1:N
     range = [j (N*2)+1-j];
     temp(j, :) = temp(j, :) + sum(DOSBlock(range , :));

end
DOSBlock = flip(temp, 1); clear temp;

figure(1), subplot(1,2,1);
contourf(1:size(DOSBlock, 1), Energy, log(DOSBlock'), 'edgecolor', 'none');
title('Density of States along DNA Strand');
xlabel('Base-Pair Number (left to right)');
ylabel('Energy (eV)');
colormap hot;
caxis([-15 10]);
axis([1 9 -6 -5.5])
%set(gca, 'FontSize', 30, 'LineWidth', 2); %<- Plot properties
xticks(1:size(DOSBlock, 1))
% ylim([-4.8 -4.5])
% caxis([-4 0])
%title([num2str(num_basepairs) '-mer'])

subplot(1,6,4), plot(gammaLcoeffs, E), axis([0 1 -6 -5.5]), title('\Gamma_L Coeffs');
subplot(1,6,5), plot(gammaRcoeffs, E), axis([0 1 -6 -5.5]), title('\Gamma_R Coeffs');
```